MAR 13 2006

WOOD, HERRON & EVANS, L.L.P.

BRUCE TITTEL
DAVID S. STALLARD
J. ROBERT CHAMBERS
GREGORY J. LUNN
KURT L. GROSSMAN
CLEMENT H. LUKEN, JR.
THOMAS J. BURGER
GREGORY F. AHRENS
WAYNE L. JACOBS
KURT A. SUMME
KEVIN G. ROONEY
KEITH R. HAUPT
THEODORE R. REMAKLUS
THOMAS W. HUMPHREY
SCOTT A. STINEBRUNER
DAVID H. BRINKMAN
BEVERLY A. LYMAN, PH.D.
KRISTI L. DAVIDSON
KATHRYN E. SMITH
P. ANDREW BLATT, Ph.D.
DAVID E. JEFFERIES

2700 CAREW TOWER
441 VINE STREET
CINCINNATI, OHIO 45202-2917
TELEPHONE: 513-241-2324
FACSIMILE: 513-241-6234
WEBSITE:: www.whepatent.com

PATENT, TRADEMARK, COPYRIGHT
AND UNFAIR COMPETITION LAW
AND RELATED LITIGATION

EDMUND P. WOOD 1923-1968
TRUMAN A. HERRON 1935-1976
EDWARD B. EVANS 1936-1971

JOSEPH R. JORDAN
C. RICHARD EBY

WILLIAM R. ALLEN, Ph.D.
JOHN PAUL DAVIS
DOUGLAS A. SCHOLER
BRETT A. SCHATZ
DAVID W. DORTON
SARAH OTTE GRABER
STEVEN W. BENINTENDI, Ph.D.
RANDALL S. JACKSON, JR.
CARL J. BRAUCH

OF COUNSEL
JOHN D. POFFENBERGER
DAVID J. JOSEPHIC
DONALD F. FREI
THOMAS W. FLYNN
J. DWIGHT POFFENBERGER, JR.
BRADLEY D. BECK

March 13, 2006

FACSIMILE COVER SHEET

To: Examiner Chrystine Pham
Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22213-1450

Fax: 571-273-8300

Enclosures:
Fax Cover Sheet containing Certificate of Facsimile Transmission (1 page)
Transmittal containing Certificate of Facsimile Transmission (1 page)
Reply Brief (including cover page - 8 pages)

From: Scott A. Stinebruner
Reg. No. 38,323

Re: U.S. Patent Application

Application No.	09/998,511
Filed:	November 30, 2001
Applicant:	Jeremy Alan Arnold et al.
Art Unit:	2192
Confirmation No.:	6560
Our Ref:	IBM/194

Pages: 10 (including cover sheet)

MESSAGE/COMMENTS
OFFICIAL

CERTIFICATE OF FACSIMILE TRANSMISSION

I hereby certify that this correspondence and the enclosures noted herein (10 total pages, including cover sheet) are being transmitted via facsimile transmission to Examiner Chrystine Pham, Mail Stop Appeal Brief- Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 at 571-273-8300 on March 13, 2006.

Judith L. Volk — March 13, 2006

Judith L. Volk — Date

MAR 13 2006

PATENT

IBM/194
Confirmation No. 6560

CERTIFICATE OF FACSIMILE TRANSMISSION

I hereby certify that this correspondence and the enclosures noted herein (10 total pages, including cover sheet) are being transmitted via facsimile transmission to Examiner Chrystine Pham, Mail Stop Appeal Brief- Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 at 571-273-8300 on March 13, 2006.

Judith L. Volk — March 13, 2006

Judith L. Volk — Date

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Jeremy Alan Arnold et al.	Art Unit:	6560
Application No.:	09/998,511	Examiner:	Chrystine Pham
Filed:	November 30, 2001	Atty. Docket No.:	IBM/194
For:	DISTRIBUTED NETWORKING USING LOGICAL PROCESSES		

TRANSMITTAL

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

We are transmitting herewith the attached:

Fax Cover Sheet containing Certificate of Facsimile Transmission (1 page)
Transmittal containing Certificate of Facsimile Transmission (1 page)
Reply Brief (including cover page - 8 pages)

If any additional fees for claims or extension of time is required or if any credit is due, please charge/credit such fees to Deposit Account No. 23-3000.

By: Scott A. Stinebruner

Scott A. Stinebruner
Reg. No. 38,323

WOOD, HERRON & EVANS, L.L.P.
2700 Carew Tower
441 Vine Street
Cincinnati, Ohio 45202
Telephone: (513) 241-2324
Facsimile: (513) 241-6234

PATENT

Attorney Docket No. IBM/194
Confirmation No. 6560

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte Jeremy Alan Arnold and John Matthew Santosuosso

Appeal No. _____
Application No. 09/998,511

REPLY BRIEF

CENTRAL FAX CENTER

MAR 13 2006

PATENT

IBM/194
Confirmation No. 6560

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Jeremy Alan Arnold et al.	Art Unit:	6560
Application No.:	09/998,511	Examiner:	Chrystine Pham
Filed:	November 30, 2001	Atty. Docket No.:	IBM/194

For: DISTRIBUTED NETWORKING USING LOGICAL PROCESSES

REPLY BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This paper is submitted in reply to the Examiner's Answer dated January 13, 2006, within the two month period for response. Consideration of the additional remarks presented below, and reversal of all of the Examiner's rejections, are respectfully requested.

From a review of the Examiner's arguments, it is apparent that the Examiner's arguments are based upon a fundamental mischaracterization of the concept of an "implementation of a method" that is "defined" in a program entity, as is used repeatedly throughout the claims. It is only through this mischaracterization that the primary reference to *Nishimura* can be read upon Applicants' claims, and as a result, this reply brief is being submitted for the purpose of clarifying this concept and correcting the Examiner's mischaracterization thereof.

Turning first to claim 1, this claim generally recites setting an inheritance breakpoint that is associated with a first program entity in an object-oriented computer program and a method identified in the first program entity. The claim further recites halting execution of the object-oriented computer program during debugging in response to reaching an implementation of the method, where that implementation of the method is defined in a second program entity that is different from and that depends from the first program entity.

As such, claim 1 requires that the inheritance breakpoint be set on a method that is associated with a method identified in one program entity, and that execution be halted upon reaching an "implementation" of that method that has been "defined" in another program entity that depends from the other program entity.

It should be apparent from Applicants' specification that an "implementation" of a method that is defined in a program entity requires that the definition of that implementation be within the program entity itself, rather than being inherited from a parent program entity. The Application, at page 8, lines 12-21, for example, describes program entities and method implementations as follows:

> A program entity with which an inheritance breakpoint may be associated may be any of a number of different program structures supported by an object-oriented programming environment within which a method may be identified and/or defined, e.g., a class, an abstract class, an interface, etc. Moreover, a program entity within which an implementation of a method may be incorporated may include any program structure capable of including program code that implements a method, typically a class. As such, the program entities within which an inheritance breakpoint is associated, and within which a method is implemented, may be related as interface-implementing class, superclass-subclass (wherein one class is a child, grandchild, great-grandchild, etc. of the other), abstract class-implementing class, etc. (*emphasis added*).

Applicants also focus on three different types of method implementations that may be defined in different, dependent program entities from that in which a method is identified. The first type of method implementation in a dependent program entity is an overridden method defined in a subclass that inherits from a parent or base class within which the same method is identified, as described in the Application at page 3, lines 5-10. Of note, this passage refers to "implementations" that are inherited from a base class, but which are not incorporated into a subclass when modified by overriding amendments in a derived class definition.

The second type of method implementation in a dependent program entity is an implementing method defined in a class that implements a public interface, as described in the Application at page 3, lines 11-22. In this type of implementation, the method is identified in a public interface, but no actual implementation of that method is provided in the interface

definition itself. Instead, as described at page 3, lines 16-18, "a programmer is required to incorporate the actual program code that implements any defined methods within the class that implements the interface."

The third type of method implementation in a dependent program entity is in an overridden method defined in a subclass that inherits from an abstract class that itself is not capable of being instantiated. This type of implementation is similar in nature to an overridden method in a subclass that inherits from a non-abstract class, as described in the Application at page 3, lines 23-28.

A common characteristic of each type of method implementation in a dependent program entity, as defined in the Application, is that the program code that actually defines the method implementation is provided within the definition for the dependent entity, rather than being inherited from a base class, public interface, abstract class or any other "parent" entity. Indeed, the Board will note that Applicants have taken great care in the claims to distinguish between the concepts of "identifying" a method and "defining" a method. Reciting that a method is "defined" in a particular program entity therefore requires more than a simple identification of the method in that program entity.

The Examiner, however, appears to take the position that an "implementation" of a method that has been "defined" in a dependent program entity, as required by claim 1, can include a method that has been defined and implemented in a parent program entity and simply inherited into the dependent program entity. However, as Applicants have chosen in both the claims and the supporting specification to articulate the concept of a method implementation that is defined in a program entity to require that the actual program code that defines the implementation be provided within that program entity, rather than being inherited from another program entity, the Examiner's interpretation of the concept of a method implementation defined in a program entity is in error.

The arguments presented by the Examiner in the Examiner's Answer are fundamentally premised on the incorrect assumption that Applicants' claims read on the situation where a breakpoint is set on a method defined in a parent class and execution is halted when that method

is executed by an object instance of a subclass that has inherited the method, but where the actual implementation of the method that is hit is defined in the parent class, but not in the subclass.

For example, the Examiner asserts at page 5, and again at pages 12-13, of the Examiner's Answer that the example set forth in page 8 of Applicants' Appeal Brief supports the Examiner's position. To the contrary, this example was set forth to explain the simple fact that if one class depends from another class (e.g., any of classes B-E that depend from class A), and a breakpoint is set on an operation (e.g., a method) defined in the parent class, that breakpoint will be hit anytime that operation is encountered in any object created from any of the dependent classes because all of the classes inherit the same operation from the parent class. The example, however, does not address the situation where an implementation of a method is defined in a dependent class (rather than defined in and inherited from the parent class). It is this latter situation that is addressed by claim 1, and *Nishimura* simply does not address this situation, as explained in detail in Applicants' Appeal Brief. Therefore, consistent with Applicants' position in the Appeal Brief, the example does in fact serve to distinguish claim 1 from *Nishimura*.

Likewise, the Examiner's arguments presented on pages 6-8 of the Examiner's Answer rely on an incorrect reading of claim 1 that presupposes that a method implementation defined in a program entity can be inherited from another program entity. The Examiner's example of a subclass A depending from a base class, where subclass A is mapped to the second program entity, and the base class is mapped to the first program entity, explicitly calls out setting breakpoints on "all functions *to be inherited in subclass A*." (*emphasis in original*). This example, however, disregards the language in claim 1 that requires that a method implementation be defined in the second program entity. In the Examiner's example, the function that is inherited is not defined in subclass A - it is instead defined in the base class and inherited by subclass A, making the function "implemented" in the base class, rather than subclass A.

The Examiner also appears to take the position on pages 8-10 of the Examiner's Answer that claim 1 is not limited to breakpoints set on overriding methods. While this is in fact the case, since claim 1 also covers in the least halting execution for implementations of methods defined in implementations of a public interface, the claim does require that the implementation of a method (of which an overridden method is an example) be defined in the second program

entity. In this regard, the Examiner states on page 10 of the Examiner's Answer that "an implementation of the method defined in a second program entity" as recited in claim 1 "reads on an inherited function." However, as described above, this claim language specifically does not read on inherited functions, contrary to the Examiner's assertions, since an inherited function is not actually defined in the second program entity.

With claim 1 properly construed to require that an implementation of a method that is defined in a program entity be actually present in the program entity, rather than being inherited from another program entity, it is evident that *Nishimura* does not disclose each and every feature of claim 1, as explained in greater detail in Applicants' Appeal Brief. Accordingly, Applicants respectfully request that the Examiner's rejection of claim 1 be reversed.

With respect to claim 8, the Examiner again mischaracterizes the concept of an implementation of a method that is defined in a program entity to include a method that is inherited from, but actually implemented and defined in another program entity. To this extent, the Board is directed to Applicants arguments above with respect to claim 1. Also, the Examiner makes an additional mischaracterization of Applicants' position in referring to claim 3. Specifically, the Examiner argues that claim 3 contradicts Applicants assertion that implementations of a method be physically found in a particular class or program entity. To the contrary, claim 3, which refers to a second implementation of a method being defined in the first program entity, supports Applicants' interpretation of the claims, since claim 3 clarifies that an implementation of a method defined in a parent program entity or class is in fact different from an implementation of a method defined in a subclass or dependent program entity. Put another way, a method that is implemented and defined in a parent program entity is not implemented and defined in the dependent program entity simply because the method is inherited by the dependent program entity. Accordingly, Applicants respectfully request that the Examiner's rejection of claim 8 be reversed for this additional reason.

With respect to claim 15, the Examiner again mischaracterizes the concept of an implementation of a method that is defined in a program entity to include a method that is inherited from, but actually implemented and defined in another program entity. To this extent, the Board is again directed to Applicants arguments above with respect to claim 1. Also, the

Examiner makes an additional mischaracterization of claim 15 on page 18 of the Examiner's Answer, arguing that the claim recites a plurality of implementations of a method, and that the plurality of implementations can include implementations that are found in the first program entity as well as the second program entity. The Examiner then concludes that, since the claim permits breakpoints to be set for implementations found in both program entities, claim 15 reads upon implementations that are found in the first program entity, and is thus anticipated by *Nishimura*.

However, whether or not "setting a breakpoint for at least a subset of the plurality of implementations", as recited in claim 15, includes setting a breakpoint on an implementation in the first (parent) program entity is immaterial, because claim 15 requires in the least that the subset of implementations upon which breakpoints are set must include at least one implementation that is defined in the second program entity. As discussed above in connection with claim 1, an implementation that is defined in a program entity does not include an implementation that is inherited from and defined in another program entity, so *Nishimura* fails to anticipate claim 15 under the same logic as applied to claim 1. Even if, *arguendo*, *Nishimura* did read upon the concept of setting a breakpoint on an implementation of a method that is defined in a parent program entity, claim 15 still requires that a breakpoint be set on at least one implementation of a method that is defined in the second program entity. Accordingly, Applicants respectfully request that the Examiner's rejection of claim 15 be reversed for this additional reason.

Finally, with respect to claims 2, 5, 9, 13, 14, 19, 21, and 23-24, the Examiner's position with respect to each of these claims is based at least in part upon the aforementioned mischaracterization. Accordingly, Applicants respectfully request that the Examiner's rejections of these claims be reversed, as these claims are based at least in part upon the concept that an implementation of a method that is defined in a program entity is actually present in the program entity, rather than being inherited from another program entity.

Reversal of the Examiner's rejections, and allowance of all claims, are therefore respectfully requested. If there are any questions regarding the foregoing, please contact the

undersigned at (513) 241-2324. Moreover, if any other charges or credits are necessary to complete this communication, please apply them to Deposit Account 23-3000.

Respectfully submitted,

WOOD, HERRON & EVANS, L.L.P.

Date: 13 MARCH 2006

By: _____

Scott A. Stinebruner
Reg. No. 38,323

2700 Carew Tower
441 Vine Street
Cincinnati, Ohio 45202
(513) 241-2324